

AFRL-IF-RS-TR-1998-79
Final Technical Report
May 1998



ENHANCING RULES IN ACTIVE DATABASES VIA RELAXATION TECHNIQUES

University of California, Los Angeles

Wesley W. Chu

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

19980618 095

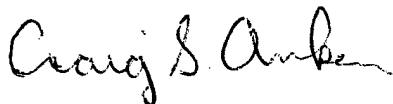
**AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK**

DTIC QUALITY INSPECTED 1

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-1998-79 has been reviewed and is approved for publication.

APPROVED:



CRAIG S. ANKEN
Project Engineer

FOR THE DIRECTOR:



NORTHROP FOWLER, III, Technical Advisor
Information Technology Division
Information Directorate

If your address has changed or if you wish to be removed from the Air Force Research Laboratory Rome Research Site mailing list, or if the addressee is no longer employed by your organization, please notify AFRL/IFTB, 525 Brooks Road, Rome, NY 13441-4505. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
<small>Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.</small>				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE May 1998	3. REPORT TYPE AND DATES COVERED Final Jun 96 - Sep 97		
4. TITLE AND SUBTITLE ENHANCING RULES IN ACTIVE DATABASES VIA RELAXATION TECHNIQUES		5. FUNDING NUMBERS C - F30602-96-1-0255 PE - 62702F PR - R427 TA - 00 WU - P8		
6. AUTHOR(S) Wesley W. Chu				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) University of California, Los Angeles Department of Computer Science 405 Hilgard Avenue Los Angeles CA 90095		8. PERFORMING ORGANIZATION REPORT NUMBER A95-3061A-00		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/ITFB 525 Brooks Road Rome NY 13441-4505		10. SPONSORING/MONITORING AGENCY REPORT NUMBER AFRL-IF-RS-TR-1998-79		
11. SUPPLEMENTARY NOTES AFRL Project Engineer: Craig S. Anken/ITFB/(315) 330-4833				
12a. DISTRIBUTION AVAILABILITY STATEMENT Approved for public release; distribution unlimited.		12b. DISTRIBUTION CODE		
13. ABSTRACT (Maximum 200 words) In most decision support systems, events may be complex and temporal in nature. Further, human cognitive and expressive processes tend to use high-level, fuzzy concepts. We developed a cooperative sentinel that can detect complex events and support ECA rules with conceptual and approximate terms. A knowledge base will be developed to classify these conceptual terms based on application context and user types. The translation high-level concepts and approximate terms to low-level rules are leveraged on the proven matured CoBase relaxation technology. The system can operate on top of conventional relational databases. A prototype of the cooperative sentinel system has been developed at UCLA which is capable of triggering on high-level ECA rules with conceptual and approximate terms.				
14. SUBJECT TERMS Cooperative rules, cooperative operators, rule relaxation, active database systems, cooperative active database systems			15. NUMBER OF PAGES 20	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED			16. PRICE CODE	
18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED		19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED		20. LIMITATION OF ABSTRACT UL

Contents

1	Introduction	1
2	Cooperative ECA Rules	1
2.1	Rules with Conceptual Terms	2
2.2	Rules with Cooperative Operators	3
2.3	Rules with Complex Events	3
3	Cooperative Active Sentinel (CoSent)	4
3.1	Cooperative Sentinel Architecture	4
3.2	Information Flow in CoSent	5
3.3	Technical Issues	7
4	Conclusion	8

1. Introduction

In support of a timely change in planning and replanning caused by disruption of flow and/or loss of infrastructure, condition monitoring and event triggering play an important role to improve the efficiency and quicken the response of the logistic pipeline. Most of the events in decision support system are complex and temporal in nature (e.g., event A and event B occur during the past week). Further, human cognitive and expressive processes tend to use high-level, fuzzy concepts. For example, consider the following two rules: *R1: If the weather is bad, then notify the commanders in the nearby affected regions.* In this rule, *weather* and *bad* are conceptual terms, *nearby* is an approximate operator. *R2: If the number of departures of large cargo carrier (e.g., C-5, C-14) becomes significantly low in the past seven days, notify the Air Mobility Command.* In this rule, the *large cargo carrier* is a conceptual term, *significantly low* is an approximate term, and *past seven days* is a temporal event.

Event Condition Action (ECA) rules are commonly used as a trigger mechanism in active databases[WC95]. Current conventional database triggering systems, e.g., Oracle and Sybase, can only support triggering based on *low-level* events such as *insert*, *update*, *delete*, etc., and therefore is not adequate for sophisticated applications. Although there are prototypes based on object-oriented databases (e.g. Hi-Pac[Cea89], Sentinel[CAMM94]) available which can support composite events, they cannot support ECA rules with conceptual and approximate terms. Further, these systems cannot interface with triggering systems in the commonly-used relational databases. To remedy these shortcomings, we developed a cooperative active sentinel system that can support *high-level* ECA rules with conceptual and approximate terms.

Our cooperative active sentinel system consists of a rule parser, a relaxation engine, an event manager, rule manager, action manager, and data source interface. *The rule parser* parses *high-level* rules and expresses in an unified internal representation – the *RuleRep*; *The relaxation engine* transforms the conceptual and approximate terms to low-level rules based on domain knowledge for triggering. *The event manager* monitors and detects composite and temporal events. The rule manager executes rules, reschedules and terminates rule execution, and displays rule execution traces. *The action manager* transforms the approximate terms into low-level conditions and invokes actions specified in the rules. *High-level* concepts are user type and application context sensitive. A knowledge base will be developed to classify conceptual terms based on application context and user types. The translation of the high-level concepts and approximate terms are leveraged on the proven and matured CoBase relaxation technology[CYC⁺96a, CYC⁺94, CC94, MC93].

The cooperative sentinel system must have a rich interface to interconnect and access information from different types of data sources, e.g., relational and object-oriented databases, legacy databases, web databases and voice inputs. It should be able to operate directly on top of conventional DBMS as well as interface with the WWW servers and CORBA interfaces. Future research in these areas are needed.

2. Cooperative ECA Rules

To improve the expressibility of ECA rules and the flexibility in specifying rule conditions, the rules used in cooperative sentinel allows conceptual terms and/or cooperative operators

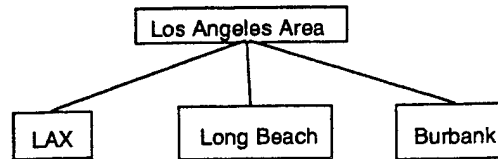


Figure 1: A TAH for Airports in Los Angeles Area.

(e.g., similar-to, near-to, or approximate) as shown in the following:

2.1. Rules with Conceptual Terms

Consider the following rule:

R1: If *wind_speed* > 40 mph and *wave_height* > 5 ft,
then notify commanders.

R1 is a precise rule which uses exact values to represent the triggering condition. Thus, the rule designers need to have detailed knowledge to specify the weather condition and notify the commanders. Further, wind speed is not a precise concept. To remedy these shortcomings, we propose to enhance rule expressibility by allowing conceptual terms in rules. As a result, *R1* can be rewritten into:

R1': If the weather is *bad*,
then notify commanders.

In this rule, “bad weather” is a conceptual term. Based on application context and domain knowledge, “bad” weather can be transformed into a range condition (e.g., *wind_speed* > 40 mph and *wave_height* > 5 ft). The domain knowledge can be represented in a tree structured knowledge representation [CC94] called *Type Abstraction Hierarchies* (TAHs), which can be automatically generated from databases by our knowledge discovery tools [CCHY96, MC93].

Using high-level active rules also reduces the number of rules in the system. The co-operative sentinel facility can derive specific low-level rules from high-level rules via the corresponding TAHs. For example, based on the TAH about Los Angeles Area Airports (Figure 1), the following rule:

R2: Notify the user if a flight from *Los Angeles Area* to New York is inserted into database.

can be rewritten into the following set of specific rules:

R2₁: Notify the user if a flight from LAX to New York is inserted into database.

R2₂: Notify the user if a flight from Burbank to New York is inserted into database.

R2₃: Notify the user if a flight from Long Beach to New York is inserted into database.

Likewise, based on the TAHs corresponding to the user type and context, a set of ECA rules can be generalized into high-level rules.

2.2. Rules with Cooperative Operators

To enhance the rule expressive power, we introduce cooperative operators such as approximate, similar-to, and near-to to specify the rule conditions. For example:

R3: If the weather turns bad,
then notify all units in that region.

can be rewritten into R3' if cooperative operators are used:

R3': If the weather turns bad, notify all units in and *near-to* that region.

Consider the following rule extracted from one of the scenarios in the ALP story board:

R4: If the aircraft has a fuel contamination problem and the aircraft type is *similar to* a 'C-5' *based on* its fuel type and fueling method,
then notify the commanders.

Note in R4 the term 'similar to' is a cooperative operator which covers a class of rules for different types of aircrafts that has similar contamination problems as C5 based on the fuel types and fueling method. Thus, it greatly increases the express power of the rule.

To process this rule, the relaxation engine translates the 'similar to' operator based on R4 to a set of exact conditions (e.g., fueling by similar design trucks from the same company instead of fuel hydrant on the airlift parking ramp). This information can be obtained from the Maintenance Database. If fuel contamination events occur, then the rule will initiate the search in the Maintenance Database and locate all aircrafts that have similar fueling methods to check for fuel filter problem.

2.3 Rules with Complex Events

The following rule is extracted from a scenario from the ALP story board:

R5: If departure rate of C-5 and C-141 within the past 7 days is *significantly low*
and
if fuel system problem rate of these aircrafts type is *extremely high* within the past 7 days
then report the aircraft type, problem, and data of occurrences to the commanders.

R5 involves two simple events, 'departure rate' and 'fuel system problem' that occur for the same type of aircraft at the same seven day period. Date and aircraft are the two parameters binding these two simple events together to form the complex temporal event. The terms *significant low* and *extremely high* used in the rules are conceptual terms which are often expressed by humans. These terms need to be translated into exact values or value range via domain knowledge which can be represented by a TAH as shown in Figure 2 (e.g., departure rate < 3/day, fuel problem > 5/day) for triggering. These conceptual terms are context and user sensitive. The relax engine transforms the cooperative ECA rules into a low-level ECA rule with exact trigger condition. The event manager will detect the occurrence of complex events and check if the condition has reached the threshold condition for initiating action.

The departure rate event may be monitored by the aircraft departure table in the Mission Database. The fuel system problem may be monitored by the Maintenance Database. The

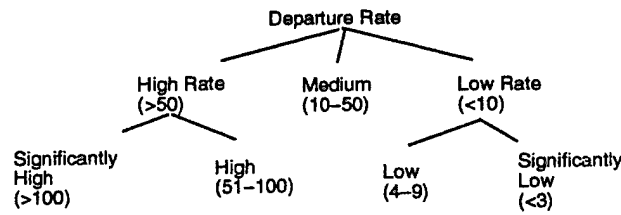


Figure 2: A TAH for Daily Aircraft Departure Rate.

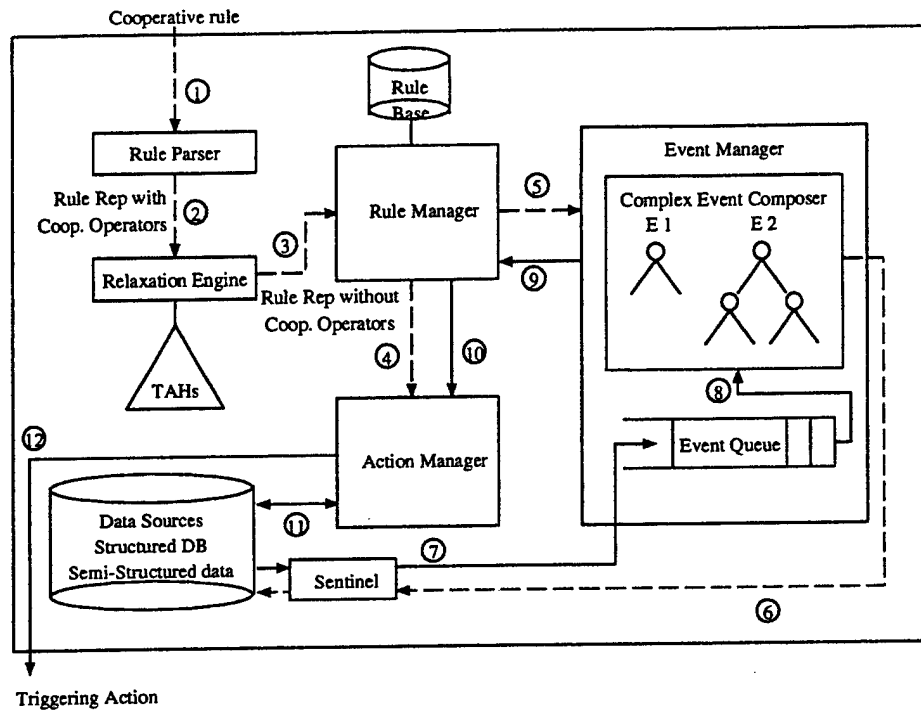


Figure 3: Cooperative Sentinel (CoSent) Architecture. The dashed line depicts installation flow, and the solid line depicts execution flow.

seven-day information can be summarized by simple database operations. Whenever the event manager detects the occurrence of this event, the rule manager will signal the event manager to query the Mission and Maintenance Databases to examine if the past seven days activities meet the specified rule conditions.

3. Cooperative Active Sentinel (CoSent)

3.1. Cooperative Sentinel Architecture

The Cooperative Sentinel (CoSent) consists of Rule Parser, Relaxation Engine, Rule Manager, Event Manager, Action Manager and underlying sentinels.

1. The Rule Parser takes a cooperative active rule and generates an internal uniform representation of the rule, RuleRep, which contains the cooperative operators.

2. The Relaxation Engine takes the RuleRep generated by the Rule Parser, translates the cooperative operators in the RuleRep into a set of exact conditions, and generates a new RuleRep without cooperative operator (all cooperative operators are translated). The Relaxation Engine also controls rule relaxation.
3. The Rule Manager is responsible for the storage, scheduling, termination management, decomposition and installation of the rules. All cooperative active rules are stored in the Rule Base. The event part is installed into the Event Manager and the action part is installed into the Action Manager.
4. The Event Manager consists of a Complex Event Composer and an Event Queue. The Complex Event Composer maintains a set of event trees, each of which represents a distinct complex event. The Event Queue buffers the incoming simple sentinel events and sequentially informs the Complex Event Composer of the occurrence of the simple events. Given a complex event from the Rule Manager, the Complex Event Composer constructs an event tree which captures the semantics of the complex event. When simple sentinel events happen, the Complex Event Composer processes them according to the event trees, evaluates the conditions if necessary, and informs the Rule Manager of the occurrence of the complex events.
5. The Action Manager is responsible for storage and dispatching of the rule actions. The Action Manager also consults database and/or relaxation system when the action requires database access or for cooperative query answering [CYC+96a].
6. The Sentinel in this architecture can be any triggering system that provides basic event notification functionality. No complex event processing is assumed.

A prototype has been constructed at UCLA to demonstrate the cooperative sentinel capabilities.

3.2. Information Flow in CoSent

There are two types of information flow in CoSent: rule installation and rule execution. Let us use (R5) as an example to illustrate the installation and execution information flow. In the installation phase, the system analyzes and decomposes all the cooperative ECA rules and installs the rule information in the Event Manager and the Action Manager. When a database event occurs, CoSent goes through the execution phase to propagate the event through the system and determines whether any rule in the Rule Base can be triggered. In the following description, step (1) through (6) describe the installation phase; step (7) through (12) represent the execution phase. These steps are also shown in Figure 3.

Installation Phase Information Flow

- (1) High-level active rule, such as (R5), is input to the CoSent.
- (2) The Rule Parser parses the high-level rule and generates an internal rule representation (RuleRep) for communication among modules. RuleRep of (R5) can be further decomposed into EventRep (E5), ConditionRep (C5) and ActionRep (A5). EventRep is the event part of the rule, e.g., in (E5) is an AND-event consists of two simple events,

insertion into aircraft departure table in the Mission Database (E5.1)
and
insertion into aircraft problem table in the Maintenance Database (E5.2)

ConditionRep represents the condition part of the rule, e.g., (C5) contains

the departure rate of C-5 and C-141 within the past 7 days is *significantly low* (C5.1)
and
the fuel system problem rate of these aircraft types is *extremely high* within the past 7 days. (C5.2)

Finally, the ActionRep represents user defined action, e.g., (A5) is

report the aircraft type, problem, and the date of occurrence to the commanders. (A5)

Note *significantly low* and *extremely high* are conceptual terms and are commonly used in human expression to specify the trigger conditions.

- (3) The Relaxation Engine translates the cooperative operators into a set of precise conditions, for example, after consulting TAHs, the condition parts, (C5.1) and (C5.2), are translated into (C5.1') and (C5.2') as follows,

the departure rate of C-5 and C-141 within the past 7 days is less than 1/day (C5.1')
and
the fuel system problem rate of these aircraft types is greater than 5/day within the past 7 day (C5.2')

which does not have any cooperative operators.

- (4) The Rule Manager installs action part, e.g., (A5), of the active rule into the Action Manager.
- (5) The Rule Manager installs event and condition parts of the active rule into the Complex Event Composer in the Event Manager. The Complex Event Manager constructs an event tree for each distinct complex event. Based on the complex event type, the input from its children nodes, and associated condition, each event node can determine the occurrence of the its corresponding complex event. For example, when the Complex Event Composer receives (E5) and (C5'), it constructs an AND-event tree with leaf nodes representing (E5.1), (C5.1') and (E5.2), (C5.2').
- (6) The Event Manager installs the simple database triggers into the triggering system. For example, (E5.1) is installed as a database trigger (T5.1) and (E5.2) is installed as a database trigger (T5.2).

Execution Phase Information Flow

- (7) When a sentinel event occurs, the database trigger, e.g., (T5.1) or (T5.2) sends an event notification to the Event Queue in the Event Manager.
- (8) The Event Queue notifies the Complex Event Composer the occurrence of a Sentinel event. Event Manager propagates the event notification through all the event trees up towards their roots.
- (9) If a root of any event tree is reached, that is all corresponding conditions are satisfied, then an event notification message with the corresponding parameter binding will be sent to the Rule Manager. For example, suppose (E5.1) had occurred and (C5.1') was satisfied, now (E5.2) occurs, after the condition (C5.2') is evaluated to true, then the complex event (E5) together with the parameter binding, aircraft type and date of occurrence are sent to the Rule Manager.
- (10) The Rule Manager schedules the execution order of the set of rules that are triggered by this event and send the parameter binding to the Action Manager. In our example, only one rule, (R5), is triggered, the parameter bindings, aircraft type and problem type are sent to the Action Manager along with the rule identification.
- (11) If the rule action needs further information, the Action Manager queries the database for necessary information. In our example, no extra information is needed.
- (12) The Action Manager invokes the user defined procedure. For example, the procedure (A5) is executed with the corresponding parameter binding and notify commander of the aircraft type, problem, and date of occurrence.

3.3. Technical Issues

The rule scheduling, termination, event history management and rule relaxation and control are some of the technical issues need to be addressed.

1. Rule Scheduling: A database event may cause the firing of multiple rules. Therefore the rule language needs to support rule firing order information (e.g. priority).
2. Rule Termination: The firing of one rule can often cause the firing of other rules. Such nested rule execution leads to the rule termination problem. One possible way to solve this problem is prohibition of database updates in the action part. Static circular rule trigger detection provides a simple but still restrictive mechanism to avoid the possible rule non-termination behavior. Although dynamic rule activation mechanisms are more powerful, they are usually much more difficult to implement and control.
3. Event History Management: Since database events are usually time sensitive, an event happened a long time ago may not be relevant in the current transaction, the rule language needs to support an event history flushing mechanism. This way, chronic events can be flushed out of the event history so that they will not affect the current transaction.

4. Rule Relaxation and Control: Similar to query relaxation, rule relaxation process also uses the TAHs from database when relaxing condition part of an cooperative active rule. The relaxation process is only done when user explicitly indicates this rule is relaxable (parser needs to recognize such indication). In this way, we only relax those rules which are specified relaxable by users and hence avoid the unexpected consequences resulted from excessive rule relaxation.

4. Conclusion

The cooperative sentinel can support temporal composite events, ECA rules with conceptual terms, and approximate operators. The system can operate on top of conventional database systems. Therefore, our system provides revolutionary technological advances in sentinel technology to meet the real-time information systems.

References

- [Camm94] S. Chakravarthy, E. Anwar, L. Maugis, and D. Mishra. Design of sentinel: An object-oriented dbms with event-based rules. *Information and Software Technology*, 36(9):559–568, 1994.
- [CC94] Wesley W. Chu and Kuorong Chiang. Abstraction of high level concepts from numerical values in databases. In *Proceedings of AAAI Workshop on Knowledge Discovery in Databases*, 1994.
- [CCHY96] Wesley W. Chu, Kuorong Chiang, Chih Cheng Hsu, and Henrick Yau. An error-based conceptual clustering method for providing approximate query answers. *Communications of ACM, Virtual Extension Edition* (<http://www.acm.org/cacm/extension>), 39(12):216–230, December 1996.
- [Cea89] S. Chakravarthy and et al. HiPAC: A research project in active, time-constrained database management (final report). Technical Report XAIT-89-02, Xerox Advanced Information Technology, Cambridge, MA, August 1989.
- [CYC+94] W.W. Chu, H. Yang, K. Chiang, B. Ribeiro, and G. Chow. GoGIS: a cooperative geographical information system. In *Proceedings of SPIE Conference on Knowledge-Based Artificial Intelligence Systems in Aerospace and Industry*, pages 53–63, Orlando, FL, April 1994.
- [CYC+96a] Wesley W. Chu, Hua Yang, Kuorong Chiang, Michael Minock, Gladys Chow, and Chris Larson. CoBase: A scalable and extensible cooperative information system. *Journal of Intelligent Information Systems*, 6(11), 1996.
- [CYC96b] W.W. Chu, H. Yang, and G. Chow. A cooperative database system (CoBase) for query relaxation. In *Proceedings of the Third International Conference on Artificial Intelligence Planning Systems*, Edinburgh, Britain, May 1996.

- [MC93] M. Merzbacher and W. W. Chu. Pattern-based clustering for database attribute values. In *Proceedings of AAAI Workshop on Knowledge Discovery*, Washington, DC, 1993.
- [SB95] J. Stillman and P. Bonissone. Developing new technologies for the ARPA-Rome Planning Initiative. *IEEE Expert*, pages 10–16, Feb 1995.
- [WC95] Jennifer Widom and Stefano Ceri. *Active Database Systems: Triggers and Rules for Advanced Database Processing*. Morgan Kaufmann, 1995.